



PATENT ABSTRACTS OF JAPAN

(11) Publication number: **11338719 A**(43) Date of publication of application: **10 . 12 . 99**

(51) Int. Cl.

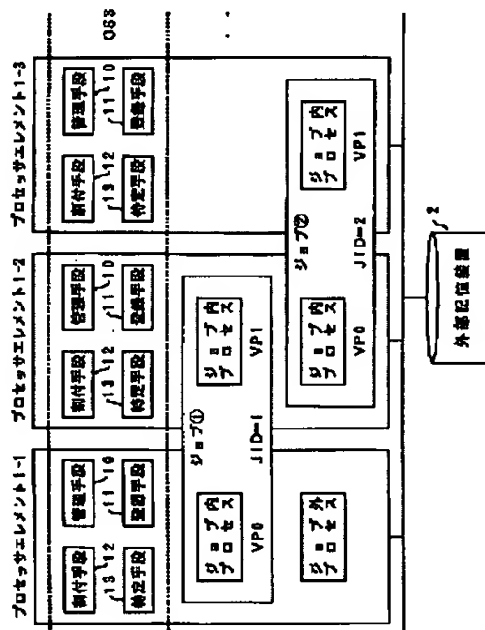
G06F 9/46**G06F 15/16**(21) Application number: **10146768**(22) Date of filing: **28 . 05 . 98**(71) Applicant: **FUJITSU LTD**(72) Inventor: **YAMASHITA KOICHIRO
WAKABAYASHI HIROHIKO**(54) **COMPUTER SYSTEM**

(57) Abstract:

PROBLEM TO BE SOLVED: To deal with the save/restoration of a job without reserving any process ID at the time of executing a job provided with plural processor elements and composed of one or plural processes.

SOLUTION: This system is provided with a managing means 10 for managing the relation of correspondence among a job ID allocated to the job, processor ID of a virtual processor element allocated to the job and processor ID of a real processor element for executing the job and an allocating means 12 for generating the process ID defined by coupling the bits of a local process ID defined in the real processor element for executing a process composing of the job when generating that process, virtual processor ID provided for that job instructed by that real processor element and job ID allocated to that job and for allocating the process ID to that process.

COPYRIGHT: (C)1999,JPO



(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平11-338719

(43) 公開日 平成11年(1999)12月10日

(51) Int.Cl.⁶

G 0 6 F 9/46
15/16

識別記号

3 6 0
3 8 0

F I

G 0 6 F 9/46
15/16

3 6 0 B
3 8 0 Z

審査請求 未請求 請求項の数 3 O L (全 12 頁)

(21) 出願番号

特願平10-146768

(22) 出願日

平成10年(1998)5月28日

(71) 出願人 000005223

富士通株式会社

神奈川県川崎市中原区上小田中4丁目1番
1号

(72) 発明者 山下 浩一郎

神奈川県川崎市中原区上小田中4丁目1番
1号 富士通株式会社内

(72) 発明者 若林 裕彦

神奈川県川崎市中原区上小田中4丁目1番
1号 富士通株式会社内

(74) 代理人 弁理士 岡田 光由 (外1名)

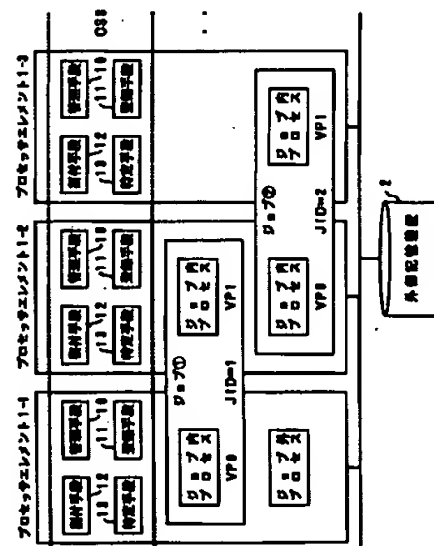
(54) 【発明の名称】 計算機システム

(57) 【要約】

【課題】本発明は、複数のプロセッサエレメントを備えて、1つ又は複数のプロセスで構成されるジョブを実行するときにあつて、プロセスIDを予約することなく、ジョブの退避・復元に対処できるようにすることを目的とする。

【解決手段】ジョブに割り付けられるジョブIDと、ジョブに割り付けられる仮想プロセッサエレメントのプロセッサIDと、ジョブを実行する実プロセッサエレメントのプロセッサIDとの対応関係を管理する管理手段10と、ジョブを構成するプロセスが生成されるときに、そのプロセスを実行する実プロセッサエレメント内で定義されるローカルプロセスIDと、その実プロセッサエレメントの指すそのジョブの持つ仮想プロセッサIDと、そのジョブに割り付けられるジョブIDとのビット結合で定義されるプロセスIDを生成して、そのプロセスに割り付ける割付手段12とを備えるように構成する。

本発明の原理構成図



【特許請求の範囲】

【請求項 1】 複数のプロセッサエレメントを備えて、1つ又は複数のプロセスで構成されるジョブを実行する計算機システムにおいて、

ジョブに割り付けられるジョブ ID と、ジョブに割り付けられる仮想プロセッサエレメントのプロセッサ ID と、ジョブを実行する実プロセッサエレメントのプロセッサ ID との対応関係を管理する管理手段と、ジョブを構成するプロセスが生成されるときに、該プロセスを実行する実プロセッサエレメント内で定義されるローカルプロセス ID と、該実プロセッサエレメントの指す該ジョブの持つ上記仮想プロセッサ ID と、該ジョブに割り付けられるジョブ ID とのビット結合で定義されるプロセス ID を生成して、該プロセスに割り付ける割付手段とを備えることを、
特徴とする計算機システム。

【請求項 2】 請求項 1 記載の計算機システムにおいて、

割付手段は、ジョブを構成しないプロセスが生成されるときに、該プロセスを実行する実プロセッサエレメント内で定義されるローカルプロセス ID と、該実プロセッサエレメントのプロセッサ ID とのビット結合で定義されるプロセス ID を生成して、該プロセスに割り付けることを、
特徴とする計算機システム。

【請求項 3】 請求項 1 又は 2 記載の計算機システムにおいて、

プロセス ID を指定して通信要求が発行されるときに、該プロセス ID と管理手段の管理データとから通信先となるプロセスを特定する特定手段を備えることを、
特徴とする計算機システム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 本発明は、複数のプロセッサエレメントを備えて、1つ又は複数のプロセスで構成されるジョブを実行する計算機システムに関し、特に、プロセス ID を予約することなく、ジョブの退避・復元に対処できるようにする計算機システムに関する。

【0002】

【従来の技術】 計算機システムでは、互いのジョブの唯一性を保証するために、ジョブに対して、互いにユニークなジョブ ID を割り付ける。

【0003】 一方、並列に拡張された計算機システムでは、ジョブは1つ又は複数のプロセスで構成されており、ジョブに属するプロセスに対しても、ジョブに属さないプロセスに対しても、互いにユニークなプロセス ID を割り付ける。

【0004】 複数のプロセッサエレメントで構成される並列プロセッサシステムでは、ジョブ ID とプロセス ID とはプロセッサエレメントを跨がって唯一性を保証し

なければならない。

【0005】 このような条件の下、プロセス群から構成されるジョブを外部記憶装置に退避した後に、他の操作を行うと、退避したジョブに属するプロセス群に割り付けられているプロセス ID が、後で生成されたプロセスのプロセス ID と重なることが起こり、これから、退避したジョブを復元する場合にプロセス ID の衝突が発生することがある。

【0006】 このプロセス ID の衝突は、退避したプロセスが多い場合や、退避中に頻繁にプロセスの生成・終了が行われる場合に、その発生の確率が高くなる。このプロセス ID の衝突を回避するために、従来では、使用の予約されているプロセス ID を登録する予約表を用意する構成を採って、プロセスを外部記憶装置に退避するときに、そのプロセスに割り付けられているプロセス ID を予約表に登録することで、プロセスの退避中に、そのプロセスのプロセス ID が後から生成されるプロセスに割り付けられないようにするという構成を採っていた。

【0007】

【発明が解決しようとする課題】 しかしながら、このような従来技術に従っていると、予約表に登録されたプロセス ID については使用できないことから、後から生成されるプロセスに割り付けられるプロセス ID の採番範囲が圧迫されるという問題点があった。

【0008】 更に、退避するプロセスの数に応じたメモリ容量を持つ予約表を用意する必要があるという問題点があった。本発明はかかる事情に鑑みてなされたものであって、複数のプロセッサエレメントを備えて、1つ又は複数のプロセスで構成されるジョブを実行する構成を採るときにあって、プロセス ID を予約することなく、ジョブの退避・復元に対処できるようにする新たな計算機システムの提供を目的とする。

【0009】

【課題を解決するための手段】 図 1 に本発明を具備する計算機システムの原理構成を図示する。この図に示すように、本発明を具備する計算機システムは、複数のプロセッサエレメント 1-i ($i=1, 2, \dots$) と外部記憶装置 2 とで構成されて、オペレーティングシステム 3 に、管理手段 10 と、登録手段 11 と、割付手段 12 と、特定手段 13 とを備える構成を採る。

【0010】 この管理手段 10 は、ジョブ (1つ又は複数のプロセスで構成される) に割り付けられるジョブ ID と、ジョブに割り付けられる仮想プロセッサエレメントのプロセッサ ID と、ジョブを実行するプロセッサエレメント 1-i の実プロセッサ ID との対応関係を管理する。

【0011】 登録手段 11 は、ジョブが生成されるときに、管理手段 10 の管理するデータを生成して、それを管理手段 10 に登録する。割付手段 12 は、プロセスが

生成されるときに、そのプロセスの識別子となるプロセスIDを生成して、それをそのプロセスに割り付ける。

【0012】特定手段13は、プロセスIDを指定して通信要求が発行されるときに、通信先となるプロセスを特定する。このように構成される本発明を具備する計算機システムでは、登録手段11の登録処理に従って、管理手段10は、ジョブIDと仮想プロセッサIDと実プロセッサIDとの対応関係を管理する。

【0013】例えば、管理手段10は、図1のプロセッサエレメント1-1,2に展開されるジョブ①については、図2に示すように、“1”というジョブID(JID)を持ち、“VP0”と“VP1”という2つの仮想プロセッサIDを持ち、実プロセッサIDとして、仮想プロセッサIDの“VP0”に割り付けられるプロセッサエレメント1-1のPE_xと、仮想プロセッサIDの“VP1”に割り付けられるプロセッサエレメント1-2のPE_yとを持つということを管理する。

【0014】また、管理手段10は、図1のプロセッサエレメント1-2,3に展開されるジョブ②については、図2に示すように、“2”というジョブID(JID)を持ち、“VP0”と“VP1”という2つの仮想プロセッサIDを持ち、実プロセッサIDとして、仮想プロセッサIDの“VP0”に割り付けられるプロセッサエレメント1-2のPE_yと、仮想プロセッサIDの“VP1”に割り付けられるプロセッサエレメント1-3のPE_zとを持つということを管理する。

【0015】一方、割付手段12は、ジョブを構成するプロセスが生成されると、そのプロセスを実行するプロセッサエレメント1-i内で定義されるローカルプロセスIDと、そのプロセッサエレメント1-iの指すそのジョブの持つ仮想プロセッサIDと、そのジョブのジョブIDとのビット結合で定義される図3(a)に示すようなプロセスIDを生成して、それをそのプロセスに割り付ける。

【0016】また、割付手段12は、ジョブを構成しないプロセスが生成されると、そのプロセスを実行するプロセッサエレメント1-i内で定義されるローカルプロセスIDと、そのプロセッサエレメント1-iの実プロセッサIDとのビット結合で定義される図3(b)に示すようなプロセスIDを生成して、それをそのプロセスに割り付ける。

【0017】ここで、図3中に示す「符号」は、ジョブを構成するプロセスであるのか否かを示す識別符号である。この本発明のプロセスIDを使うことで、ジョブを外部記憶装置2に退避した後に、他の操作を行うことで、後で生成されたプロセスに対して、退避したジョブの持つプロセスに割り付けられているものと同一のローカルプロセスIDが割り付けられるようなことが起きても、何も不都合は起こらない。

【0018】また、この本発明のプロセスIDを使うこ

とで、ジョブをプロセッサエレメント1-i間で移動するときに、移動先に、そのジョブの持つプロセスに割り付けられているものと同一のローカルプロセスIDを持つプロセスが展開されているようなことがあっても、何も不都合は起こらない。

【0019】すなわち、本発明のプロセスIDは、従来技術で使われているローカルプロセスIDを拡張したデータ構造を持つので、この拡張部分のデータ構造の違いにより、同一のローカルプロセスIDを持つようなことがあってもプロセスを識別することができるのである。

【0020】この本発明のプロセスIDを用いるときに、プロセスIDを指定して通信要求が発行されると、特定手段13は、そのプロセスIDの持つ「符号」がジョブを構成しないプロセスであることを示しているときには、そのプロセスIDの持つ実プロセッサIDの指すプロセッサエレメント1-iに展開される、そのプロセスIDの持つローカルプロセスIDの指すプロセスを通信先として特定する。

【0021】一方、特定手段13は、通信要求先として指定されるプロセスIDの持つ「符号」がジョブを構成するプロセスであることを示しているときには、そのプロセスIDの持つジョブID/仮想プロセッサIDを検索キーにして管理手段10を検索することで実プロセッサIDを特定して、その特定した実プロセッサIDの指すプロセッサエレメント1-iに展開される、そのプロセスIDの持つローカルプロセスIDの指すプロセスを通信先として特定する。

【0022】このようにして、本発明のプロセスIDを用いるときには、管理手段10の管理データを参照することで、簡単に通信要求先のプロセスを特定できるようになる。

【0023】

【発明の実施の形態】以下、実施の形態に従って本発明を詳細に説明する。図4に本発明を具備する計算機システムの一実施例を図示する。

【0024】この実施例に従う本発明を具備する計算機システムは、PE_x、PE_y、PE_z、PE_αという4台のプロセッサエレメントと、外部記憶装置2とを備えている。

【0025】各プロセッサエレメント上では、オペレーティングシステムが動作しており、システムの運用のために、単一あるいは複数のプロセス群から構成される複数のジョブと、ジョブを構成しない複数のプロセスとが動作している。

【0026】各プロセッサエレメントには、オペレーティングシステムから通知されるジョブ管理表100が展開される。このジョブ管理表100は、ジョブに割り付けられるジョブIDと、ジョブに割り付けられる仮想プロセッサIDと、ジョブを実行するプロセッサエレメントの実プロセッサIDとの対応関係を管理する。

10

20

30

40

50

【0027】オペレーティングシステムは、ジョブがシステムに投入されると、そのジョブに対してユニークなジョブIDを割り付けるとともに、そのジョブを実行するプロセッサエレメントに対して、相対的な通番で定義される仮想プロセッサIDを割り付け、それを各プロセッサエレメントに展開されるジョブ管理表100に登録する。ここで、ジョブIDと仮想プロセッサIDとは、そのジョブが存在している間是不変値として扱われる。

【0028】図4の実施例で説明するならば、システムに投入されるジョブ①に対して、“1”というジョブIDを割り付けるとともに、そのジョブ①を実行するプロセッサエレメントPE_xに対して“VP0”という仮想プロセッサIDを割り付け、そのジョブ①を実行するプロセッサエレメントPE_yに対して“VP1”という仮想プロセッサIDを割り付けて、それらをジョブ管理表100に登録するのである。

【0029】また、システムに投入されるジョブ②に対して、“2”というジョブIDを割り付けるとともに、そのジョブ②を実行するプロセッサエレメントPE_yに対して“VP0”という仮想プロセッサIDを割り付け、そのジョブ②を実行するプロセッサエレメントPE_zに対して“VP1”という仮想プロセッサIDを割り付けて、それらをジョブ管理表100に登録するのである。

【0030】そして、オペレーティングシステムは、ジョブが操作されると、その操作に応じて、必要に応じてジョブ管理表100の管理データを書き換えていくよう処理する。

【0031】例えば、図4に示したジョブ②が図5に示すように移動されることで、ジョブ②を実行するプロセッサエレメントがPE_y/PE_zからPE_z/PE_αに移動すると、ジョブ②を実行するプロセッサエレメントPE_zに対して“VP0”という仮想プロセッサIDを割り付け、ジョブ②を実行するプロセッサエレメントPE_αに対して“VP1”という仮想プロセッサIDを割り付けて、それらに従ってジョブ管理表100の管理データを書き換えていくのである。

【0032】なお、このジョブ管理表100は、標準のUNIXシステムのシェルが持つジョブの管理表を配列変数を用いて並列に拡張したものである。更に、オペレーティングシステムは、プロセスが生成されると、図6の処理フローに従って、そのプロセスに対して、本発明に特徴的なプロセスIDを割り付ける。

【0033】次に、図6に示す処理フローに従って、このプロセスIDの割付処理について説明する。オペレーティングシステムは、プロセスが生成されると、図6の処理フローに示すように、先ず最初に、ステップ1で、生成されたジョブに対して、単一プロセッサ構成を採る標準のUNIXシステムで用いられる16ビットのプロセスID（以下、ローカルプロセスIDと称する）を生

成する。

【0034】続いて、ステップ2で、生成されたプロセスがジョブを構成するものであるのか否かを判断して、ジョブを構成することを判断するときには、ステップ3に進んで、生成されたプロセスの属するジョブのジョブIDを特定し、続くステップ4で、生成されたプロセスを実行するプロセッサエレメントの実プロセッサIDを特定する。

【0035】続いて、ステップ5で、ステップ3で特定したジョブIDと、ステップ4で特定した実プロセッサIDとを検索キーにして、ジョブ管理表100を検索することで、生成されたプロセスに対応付けられる仮想プロセッサIDを特定する。

【0036】続いて、ステップ6で、ステップ1で生成したローカルプロセスIDと、ステップ3で特定したジョブIDと、ステップ5で特定した仮想プロセッサIDとをビット結合するとともに、その先頭にジョブを構成するプロセスのプロセスIDであることを示すフラグ値“1”（ジョブ判定フラグ）を付加することで、図7（a）に示すような32ビットのプロセスIDを生成して、それを生成されたプロセスに割り付ける。

【0037】一方、ステップ2で、生成されたプロセスがジョブを構成しないことを判断するときには、ステップ7に進んで、生成されたプロセスを実行するプロセッサエレメントの実プロセッサIDを特定する。

【0038】続いて、ステップ8で、ステップ1で生成したローカルプロセスIDと、ステップ7で特定した実プロセッサIDとをビット結合するとともに、その先頭にジョブを構成しないプロセスのプロセスIDであることを示すフラグ値“0”（ジョブ判定フラグ）を付加することで、図7（b）に示すような32ビットのプロセスIDを生成して、それを生成されたプロセスに割り付ける。

【0039】このようにして、単一プロセッサ構成を採る標準のUNIXシステムでは、プロセスに対して、図7（c）に示すような16ビットのプロセスID（本発明に言うローカルプロセスID）を割り付ける構成を採るのに対して、本発明を具備する計算機システムでは、並列プロセッサシステムを前提とするときに、UNIXシステムの豊富な資産を引き継ぐための互換性を考慮して、下位2バイトに、UNIXシステムで割り付けるローカルプロセスIDを持つとともに、上位2バイトに、ジョブに属するプロセスのときには、ジョブ判定フラグ/ジョブID/仮想プロセッサIDを持ち、ジョブに属さないプロセスのときには、ジョブ判定フラグ/実プロセッサIDを持つプロセスIDを割り付ける構成を採るのである。

【0040】このようにして、例えば、図4に示すジョブ①の持つプロセッサエレメントPE_xに展開されるプロセスに対して、

10

20

30

40

50

[1, 1, VP0, aaaa]

但し、“aaaa”はローカルプロセスID
という32ビットのプロセスIDが割り付けられ、図4
に示すジョブ①の持つプロセッサエレメントPEyに展
開されるプロセスに対して、

[1, 1, VP1, bbbb]

但し、“bbbb”はローカルプロセスID
という32ビットのプロセスIDが割り付けられ、図4
に示すプロセッサエレメントPExに展開されるジョブ
外プロセスに対して、

[0, PEx, cccc]

但し、“cccc”はローカルプロセスID
という32ビットのプロセスIDが割り付けられること
になる。

【0041】このプロセスIDを使うことで、本発明を
具備する計算機システムでは、ジョブを外部記憶装置2
に退避した後に、他の操作を行うことで、後で生成され
たプロセスに対して、退避したジョブの持つプロセスに
割り付けられているものと同一のローカルプロセスID
が割り付けられるようなことが起きても、何も不都合は
起こらない。

【0042】また、このプロセスIDを使うことで、本
発明を具備する計算機システムでは、ジョブをプロセッ
サエレメント間で移動するときに、移動先に、そのジョ
ブの持つプロセスに割り付けられているものと同一のロ
ーカルプロセスIDを持つプロセスが展開されているよ
うなことがあっても、何も不都合は起こらない。

【0043】すなわち、本発明を具備する計算機シス
テムで用いるプロセスIDは、標準のUNIXシステムで
使われているローカルプロセスIDを拡張したデータ構
造を持つので、この拡張部分のデータ構造の違いによ
り、同一のローカルプロセスIDを持つようなことがあ
ってもプロセスを識別することができるのである。

【0044】オペレーティングシステムは、ジョブ内プ
ロセス又はジョブ外プロセスから、プロセスIDを指定
して通信要求が発行されると、図8の処理フローに従っ
て、通信要求先のプロセスを特定する処理を行う。

【0045】次に、この通信要求先の特定処理について
説明する。オペレーティングシステムは、プロセスID
を指定して通信要求が発行されると、図8の処理フロー
に示すように、まず最初に、ステップ1で、指定される
プロセスIDの先頭の1ビットを読み取る。

【0046】続いて、ステップ2で、読み取ったビット
値から、指定されるプロセスIDの指すプロセスがジョ
ブ内プロセスであるのか、ジョブ外プロセスであるのか
をチェックして、ジョブ内プロセスであることを判断す
るときには、ステップ3に進んで、ビット演算による切
出処理を実行することで、指定されるプロセスIDか
ら、ジョブID/仮想プロセッサID/ローカルプロセ
スIDを切り出す。

【0047】続いて、ステップ4で、切り出したジョブ
ID及び仮想プロセッサIDを検索キーとしてジョブ管
理表100を検索することで、実プロセッサIDを特定
し、続くステップ5で、特定した実プロセッサIDの指
すプロセッサエレメントに展開される、ステップ3で切
り出したローカルプロセスIDの指すプロセスを通信先
として特定する。

【0048】このようにして、通信要求先として指定さ
れるプロセスIDの指すプロセスがジョブ内プロセスで
あるときには、図9に示すように、指定されるプロセス
IDとジョブ管理表100の管理データとから通信先を
特定するのである。

【0049】一方、ステップ2で、読み取ったビット値
から、指定されるプロセスIDの指すプロセスがジョブ
外プロセスであることを判断するときには、ステップ6
に進んで、ビット演算による切出処理を実行すること
で、指定されるプロセスIDから、実プロセッサID/
ローカルプロセスIDを切り出す。

【0050】続いて、ステップ7で、切り出した実プロ
セッサIDの指すプロセッサエレメントに展開される、
切り出したローカルプロセスIDの指すプロセスを通信
先として特定する。

【0051】このようにして、通信要求先として指定さ
れるプロセスIDの指すプロセスがジョブ外プロセスで
あるときには、図10に示すように、指定されるプロセ
スIDから通信先を特定するのである。

【0052】通信要求先として指定されるプロセスID
の指すプロセスがジョブ内プロセスであるときの通信先
の特定処理は、5τのマシンサイクルで実行できるの
で、高速に通信先を特定することが可能である。

【0053】すなわち、この通信先の特定処理は、先頭
の1ビットのビット値（ジョブ判定フラグ）を参照する
ことで、指定されるプロセスIDがジョブに属するの
かを判断して、ジョブに属することを判断するときには、
ジョブ管理表100を検索することで実行されることにな
るが、先頭の1ビットのビット値が“1”か“0”で
あるのかは、3τで実行される比較処理により実行さ
れ、ジョブ管理表100の検索は、2τで実行される2
次元配列の指定処理により実行されるので、合計5τの
マシンサイクルで実行できるのである。

【0054】上述したように、本発明を具備する計算機
システムで用いるプロセスIDは、単一プロセッサ構成
を採る標準のUNIXシステムで用いられるプロセスID
を拡張したものとなっている。

【0055】従って、標準のUNIXシステムの豊富な
資産を引き継ぐことが可能となる。更に、単一プロセッ
サ構成を採る標準のUNIXシステムから、本発明を具
備する並列プロセッサ構成の計算機システムへのマイグ
レード操作（移動操作）が可能になるとともに、本発明
を具備する並列プロセッサ構成の計算機システムから、

単一プロセッサ構成を採る標準のUNIXシステムへのマイグレード操作（移動操作）が可能になるという特徴を有する。

【0056】すなわち、本発明を具備する計算機システムにプロセッサエレメントが追加されるときに、既存のUNIXシステムで動作するプロセスのプロセスIDを拡張（図7に示すように上位2バイトを付加する）することで、その既存のUNIXシステムで動作するプロセスを本発明を具備する計算機システムに移動させることが可能になる。これにより、システムを簡単に拡張できるようにする。

【0057】また、本発明を具備する計算機システムのプロセッサエレメントの1つがダウンするときに、そのプロセッサエレメントで動作するプロセスのプロセスIDの上位2バイトを削除（移動先が無視するときには削除する必要はない）することで、そのプロセッサエレメントで動作するプロセスを標準のUNIXシステムに移動させることが可能になる。これにより、システムダウンに簡単に対処できるようになる。

【0058】図11に、本発明を具備する計算機システムで実行する運用処理の処理フロー、図12に、本発明を具備する計算機システムと連携する、単一プロセッサ構成を採る標準のUNIXシステムで実行する運用処理の処理フローを図示する。

【0059】次に、この処理フローについて説明する。本発明を具備する計算機システムは、起動されると、図11の処理フローで示す処理に入って、ジョブ（プロセス）に対する操作に従って、ユーザによりジョブ（プロセス）が投入されると、ローカルプロセスIDを採番し、ジョブ判定フラグ/ジョブID/仮想プロセッサIDや、ジョブ判定フラグ/実プロセッサIDという属性情報を適宜配置し結合し、それらとローカルプロセスIDとを結合することで、投入されるジョブ（プロセス）に対して、32ビットのプロセスIDを割り付けていく。

【0060】一方、ジョブ（プロセス）に対する操作に従って、ユーザによりシグナルが発行されたり、プロセス間の通信指示が発行されると、ビット演算により属性情報を抽出し、プロセスIDから処理対象のプロセスを特定して、それに対する制御や処理を行う。

【0061】一方、ジョブ（プロセス）に対する操作に従って、外部記憶装置2への退避や外部記憶装置2からの復元が行われると、外部記憶装置2への退避のときには、ユーザの指示に従ってジョブ（プロセス）を外部記憶装置2に退避させた後、ユーザの指示に従って他ジョブを投入してシステムを再起動する。また、外部記憶装置2からの復元のときには、ユーザからの指示が非移動復元（元のプロセッサエレメントに戻す復元）であるのか移動復元であるのかを判断して、非移動復元のときには、ビット演算により属性情報を抽出し、プロセスID

から処理対象のプロセスを特定して、それに対する制御や処理を行い、移動復元のときには、移動先が自システムであるのか標準のUNIXシステムであるのかを判断して、移動先が自システムのときには、ビット演算により属性情報を抽出し、プロセスIDから処理対象のプロセスを特定して、それに対する制御や処理を行う。

【0062】一方、ジョブ（プロセス）に対する操作に従って、ユーザによりマイグレード操作（移動操作）が行われると、移動先が自システムであるのか標準のUNIXシステムであるのかを判断して、移動先が自システムのときには、ビット演算により属性情報を抽出し、プロセスIDから処理対象のプロセスを特定して、それに対する制御や処理を行う。

【0063】この処理構成を採るときに、移動先が標準のUNIXシステムであるときには、図11及び図12のAに示すように、移動対象となるジョブ（プロセス）を標準のUNIXシステムへ移動させる。このとき、標準のUNIXシステムでは、32ビットのプロセスIDの内、下位16ビットのプロセスID部分のみをプロセスIDとして抽出するので、移動対象となるジョブ（プロセス）をそのまま標準のUNIXシステムへ移動させていくことになる。

【0064】また、本発明を具備する計算機システムと連携するUNIXシステムは、起動されると、図12の処理フローに示す処理に入って、ジョブ（プロセス）に対する操作に従って、ユーザによりジョブ（プロセス）が投入されると、図7（c）に示した16ビットのプロセスIDを採番することで、投入されるジョブ（プロセス）に対して、16ビットのプロセスIDを割り付けていく。

【0065】一方、ジョブ（プロセス）に対する操作に従って、ユーザによりシグナルが発行されたり、プロセス間の通信指示が発行されると、16ビットのプロセスIDから処理対象のプロセスを特定して、それに対する制御や処理を行う。

【0066】一方、ジョブ（プロセス）に対する操作に従って、ユーザによりマイグレード操作（移動操作）が行われると、図11及び図12のBに示すように、移動対象となるジョブ（プロセス）を本発明を具備する計算機システムへ移動させ、そのとき、16ビットのジョブ判定フラグ/ジョブID/仮想プロセッサIDや、ジョブ判定フラグ/実プロセッサIDという属性情報を適宜配置し結合し、それらとUNIXシステムで付加された16ビットのプロセスIDとを結合することで、移動されるジョブ（プロセス）に対して、32ビットのプロセスIDを割り付けていく。

【0067】このようにして、本発明を具備する計算機システムは、単一プロセッサ構成を採る標準のUNIXシステムの豊富な資産を引き継ぎながら、データ処理を行うことができるのである。

【0068】この実施例では、ジョブに属するプロセスとジョブに属さないプロセスとを区別するという構成を採ったが、ジョブに属さないプロセスについても、ジョブを構成するものと見なす方法を採用することも可能である。この方法を採用場合には、プロセスに割り付けられるプロセスIDは、図7(a)に示すものに統一されることになる。

【0069】

【発明の効果】以上説明したように、本発明によれば、並列プロセッサ構成を採る計算機システムにあって、標準のUNIXシステムの豊富な資産をそのまま引き継ぎながら、プロセスIDを予約することなく、ジョブの退避・復元に対処できるようになる。

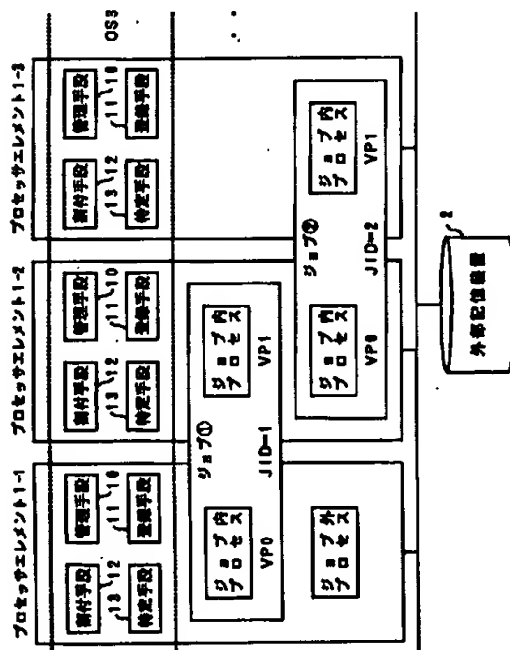
【図面の簡単な説明】

【図1】本発明の原理構成図である。

【図2】管理手段の管理するデータの説明図である。

【図3】本発明のプロセスIDの説明図である。

【図1】



【図4】本発明の一実施例である。

【図5】本発明の一実施例である。

【図6】プロセスIDの割付処理の処理フローである。

【図7】本発明のプロセスIDの一実施例である。

【図8】通信先の特定処理の処理フローである。

【図9】通信先の特定処理の説明図である。

【図10】通信先の特定処理の説明図である。

【図11】本発明の運用処理の説明図である。

【図12】本発明の運用処理の説明図である。

【符号の説明】

- 1 プロセッサエレメント
- 2 外部記憶装置
- 3 オペレーティングシステム
- 10 管理手段
- 11 登録手段
- 12 割付手段
- 13 特定手段

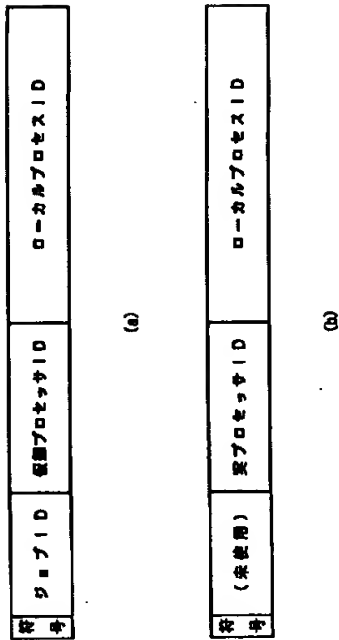
【図2】

管理手段の管理するデータの説明図

JID	VP0	VP1	VP2	...
1	PE _x	PE _y	—	...
2	PE _y	PE _z	—	...
.
.
.
.
.

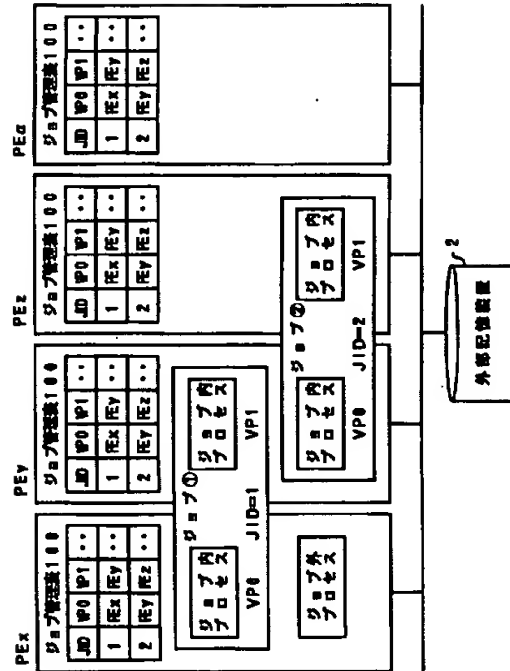
【図3】

本発明のプロセスIDの説明図



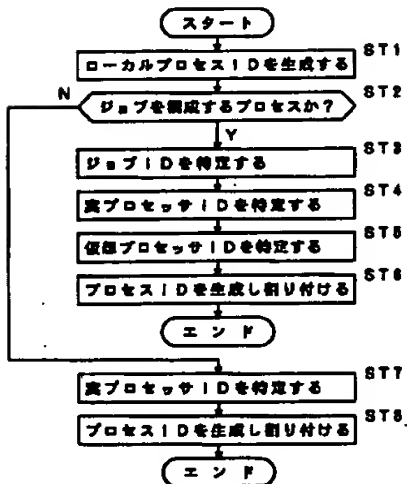
【図4】

本発明の一実施例

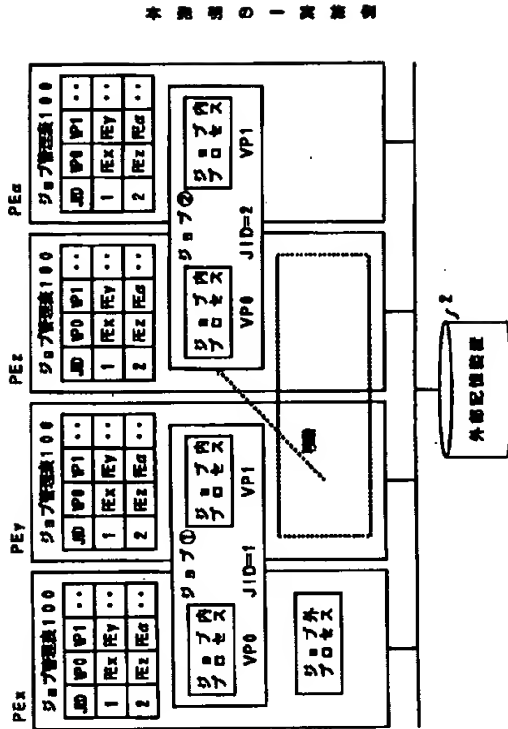


【図6】

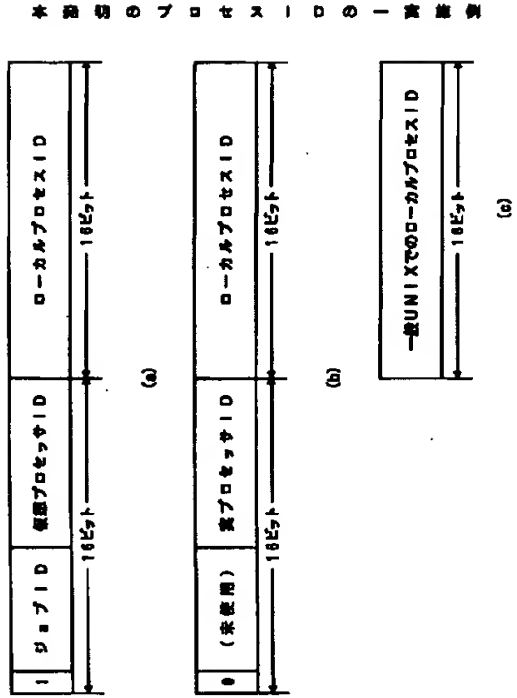
プロセスIDの割付処理の処理フロー



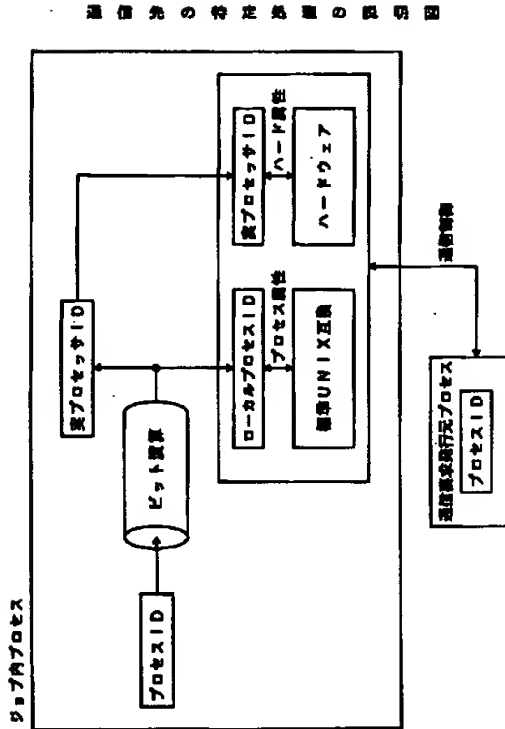
【図5】



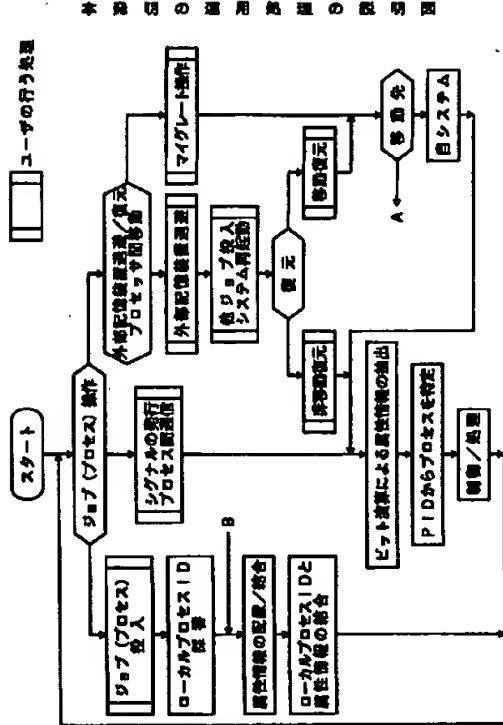
【図7】



【図10】



【図11】



【図12】

本発明の適用処理の説明図

